ДЕТСКАЯ ОНЛАЙН-ШКОЛА ПРОГРАММИРОВАНИЯ
HELLO WORLD

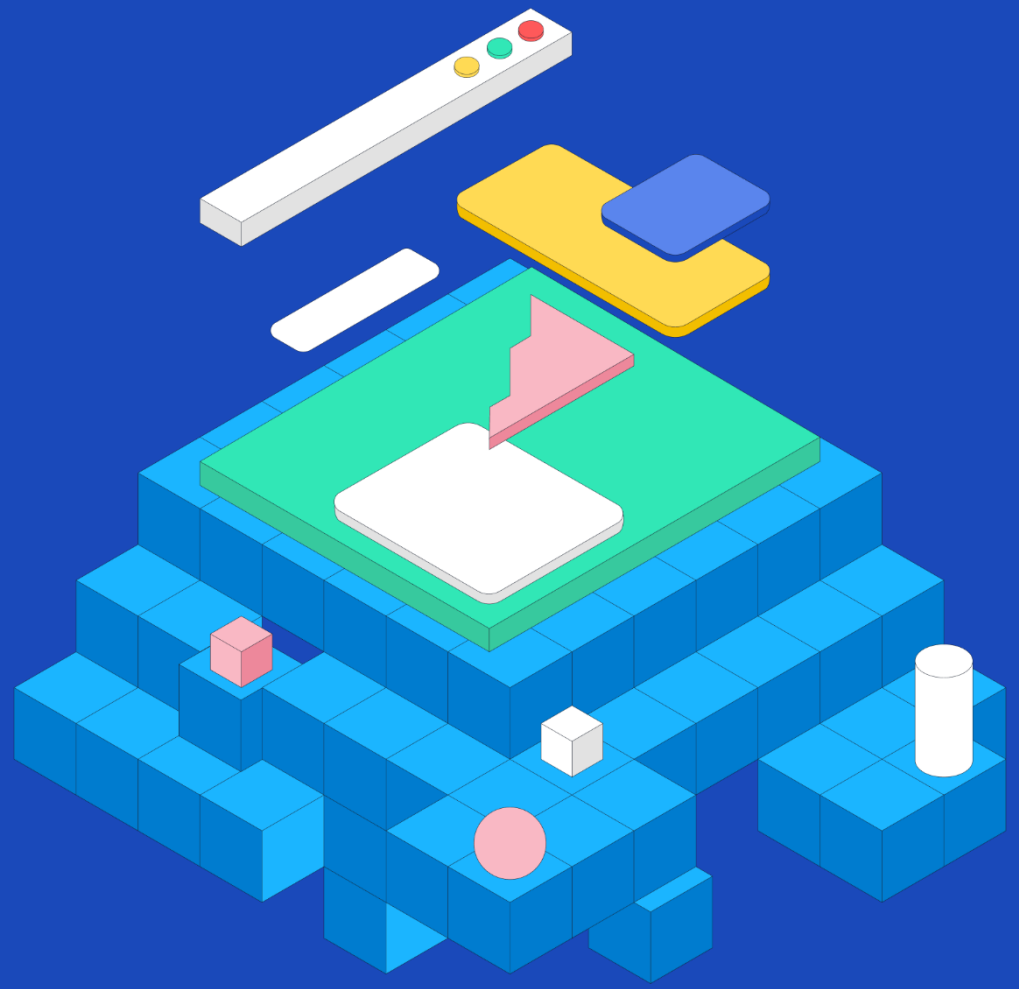
#ШПАРГАЛОЧКИ

СОЗДАНИЕ САЙТОВ FRONT-END РАЗРАБОТКА

Дополнительный уровень

Материалы подготовлены отделом методической разработки

Больше полезных материалов и общения в нашем комьюнити в Telegram: https://t.me/hw_school



FLUX архитектура (часть 2)

FLUX

FLUX - это архитектурный подход к программированию, согласно которому обновление состояния компонентов приложения не должно происходить само по себе.

Например, когда пользователь вводит текст в input, UI (пользовательский интерфейс) должен перерисовываться. Но эта перерисовка (обновление) должна происходить через сохранение введенных данных в хранилище и отрисовку их через **state**.

onChange

Чтобы отправка данных из input происходила при каждом действии (даже при введении одной буквы), нужно **отслеживать** изменения.

onChange - это обработчик событий **input**, который срабатывает при изменении поля ввода. Он может вызывать функцию, которая будет сохранять содержимое **input** и перерисовывать страницу

onChange

В **state.js**:

```
export let onPostChange = (text) => {
        state.profilePage.newPostText = text
        rerenderTree(state)
}
```

Теперь **onPostChange** нужно импортировать в render.js, App.js, Profile.js

onChange

В **render.js**:

```
<App state={state} onPostCha nge={onPostChange} addPost={addPost}
sendMessage={sendMessage} />
```

В **App.js**:

```
<Route  exact path='/' render={()=><Profile ... onPostChange={props.onPostChange} />} />
```

В **Profie.js**:

```
<Posts ... onPostChange={props.onPostChange} ... />
```

onChange

Остается прописать **onChange** в input:

<input onChange={onPostChange} … />

И создать еще одну функцию **onPostChange**, которая будет вызывать одноименную функцию из **state.js**:

```
let onPostChange = () => {
        props.onPostChange(postText.current.value)
}
```